GRASP: Greedy Randomized Adaptive Search Procedures

A metaheuristic for combinatorial optimization

Maurício G. C. Resende

Algorithms & Optimization Research Dept.

AT&T Labs Research

Florham Park, New Jersey

mgcr@research.att.com

http://www.research.att.com/~mgcr

May 2000

ECCO'2000 – Capri, Italy

Outline

- Introduction
  - combinatorial optimization & local search
  - random multi-start local search
  - greedy and semi-greedy algorithms
- A basic (standard) GRASP
- Enhancements to the basic GRASP
  - enhancements to local search
  - asymptotic behavior
  - automatic choice of RCL parameter α
  - use of long-term memory
  - GRASP in hybrid metaheuristics
  - parallel GRASP
- Survey of applications in literature
  - operations research & computer science
  - industrial

Combinatorial Optimization

- Given:
  - discrete set of solutions X
  - objective function $f(x): x \in X \rightarrow R$
- Objective:
  - find $x \in X : f(x) \leq f(y), \forall y \in X$

Origins

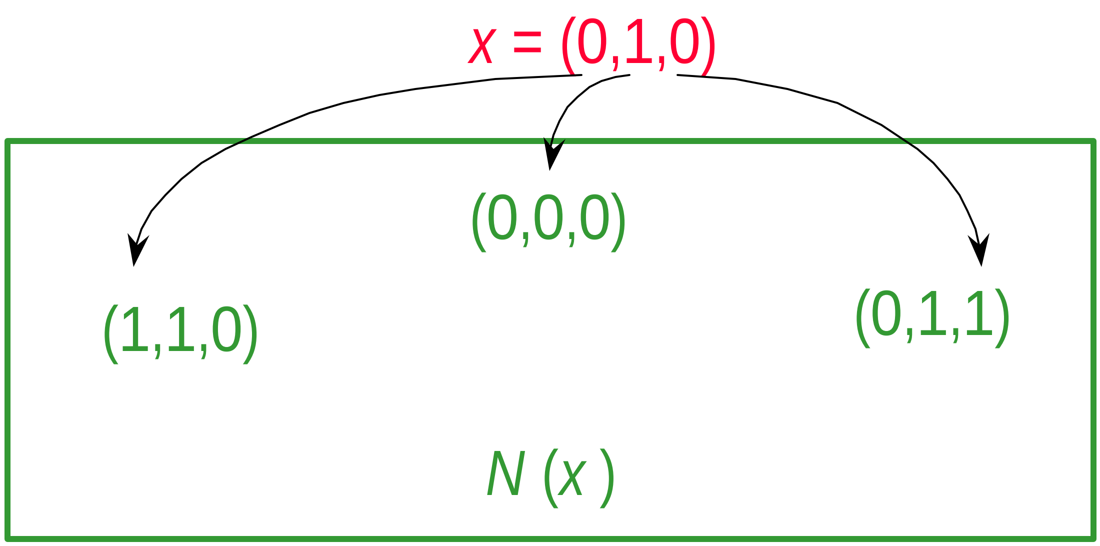
- Probabilistic algorithm (GRASP) for difficult set covering problems [Feo & R., 1989]
- GRASP was related to previous work, e.g.:
  - random multistart local search [e.g. Lin & Kernighan, 1973]
  - semi-greedy heuristics [e.g. Hart & Shogan, 1987]

Local Search

- To define local search, one needs to specify a local neighborhood structure.
- Given a solution *x* , the elements of the neighborhood *N* (*x*) of *x* are those solutions *y* that can be obtained by applying an elementary modification (often called a move) to *x*.

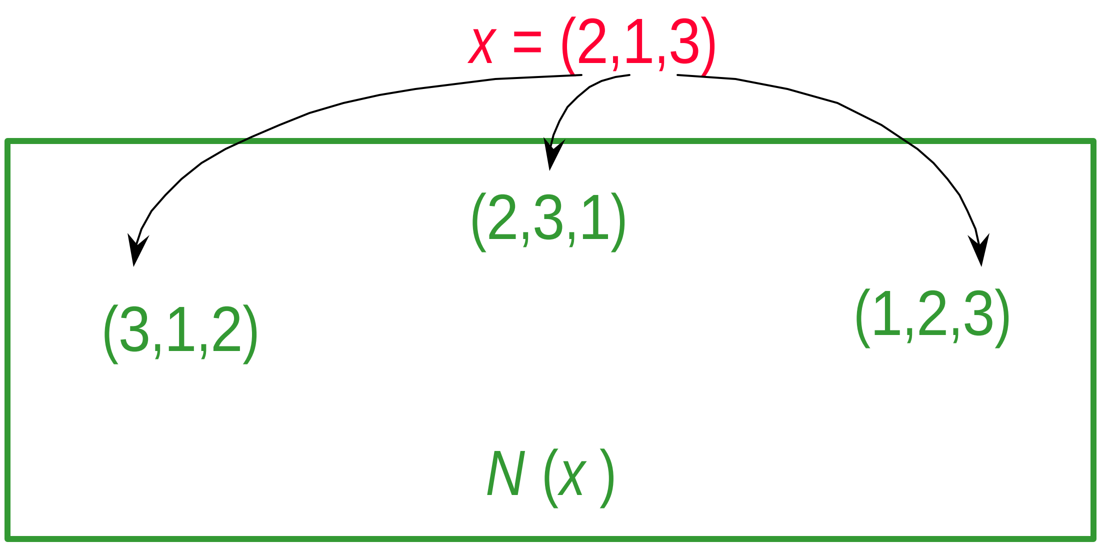
Local Search Neighborhoods

- Consider $x = (0,1,0)$ and the 1-flip neighborhood of a 0/1 array.



Local Search Neighborhoods

- Consider x = (2,1,3) and the 2-swap neighborhood of a permutation array.



Local Search

- Local search: Given an initial solution x_0, a neighborhood $N(x)$, and function $f(x)$ to be minimized:

check for better solution in neighborhood of x

$x = x_0$;

while ($\exists\, y \in N(x) \mid f(y) < f(x)$){

   $x = y$;

}

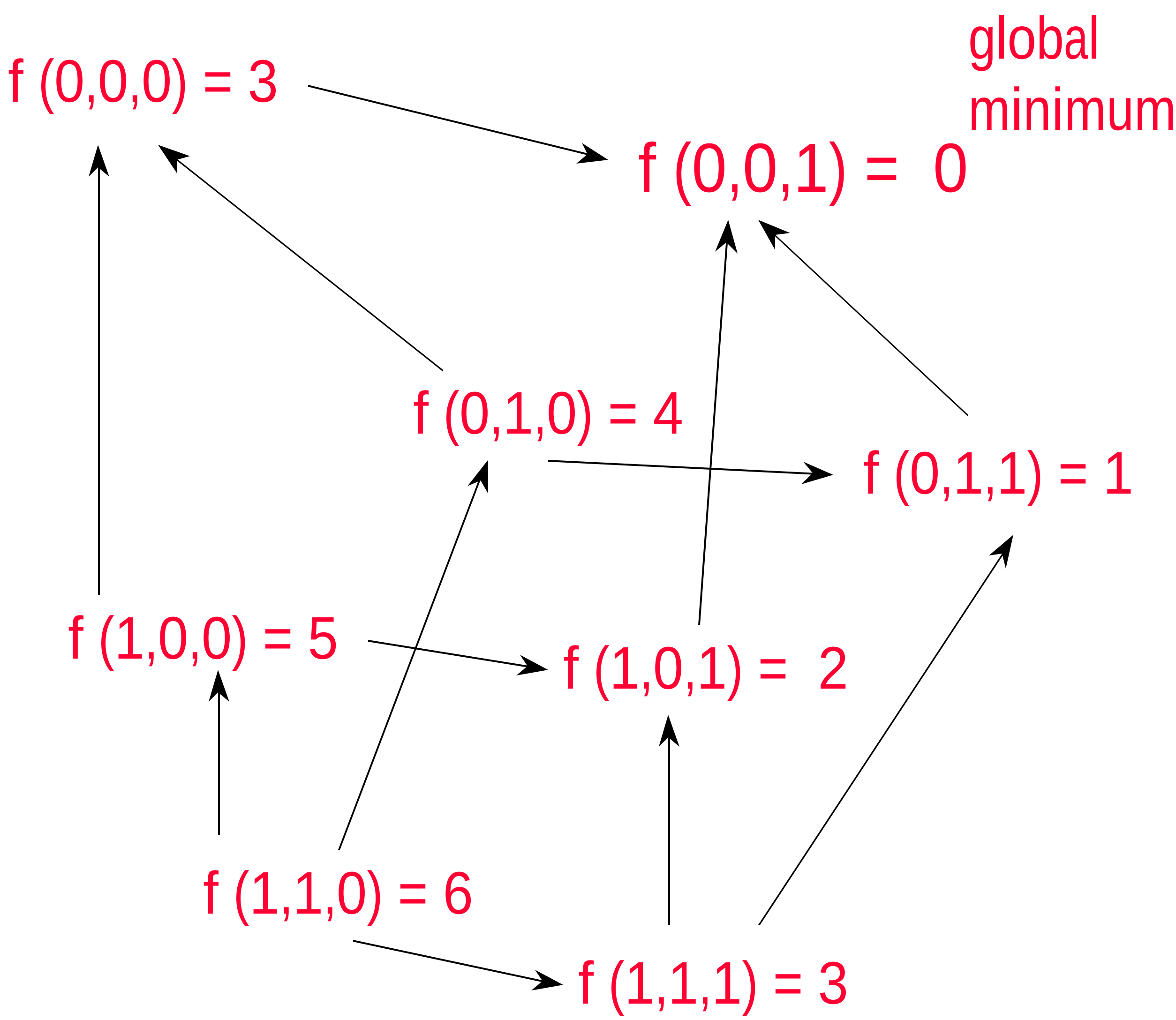
move to better solution y

At the end, x is a local minimum of $f(x)$.

Time complexity of local search can be exponential.

Local Search

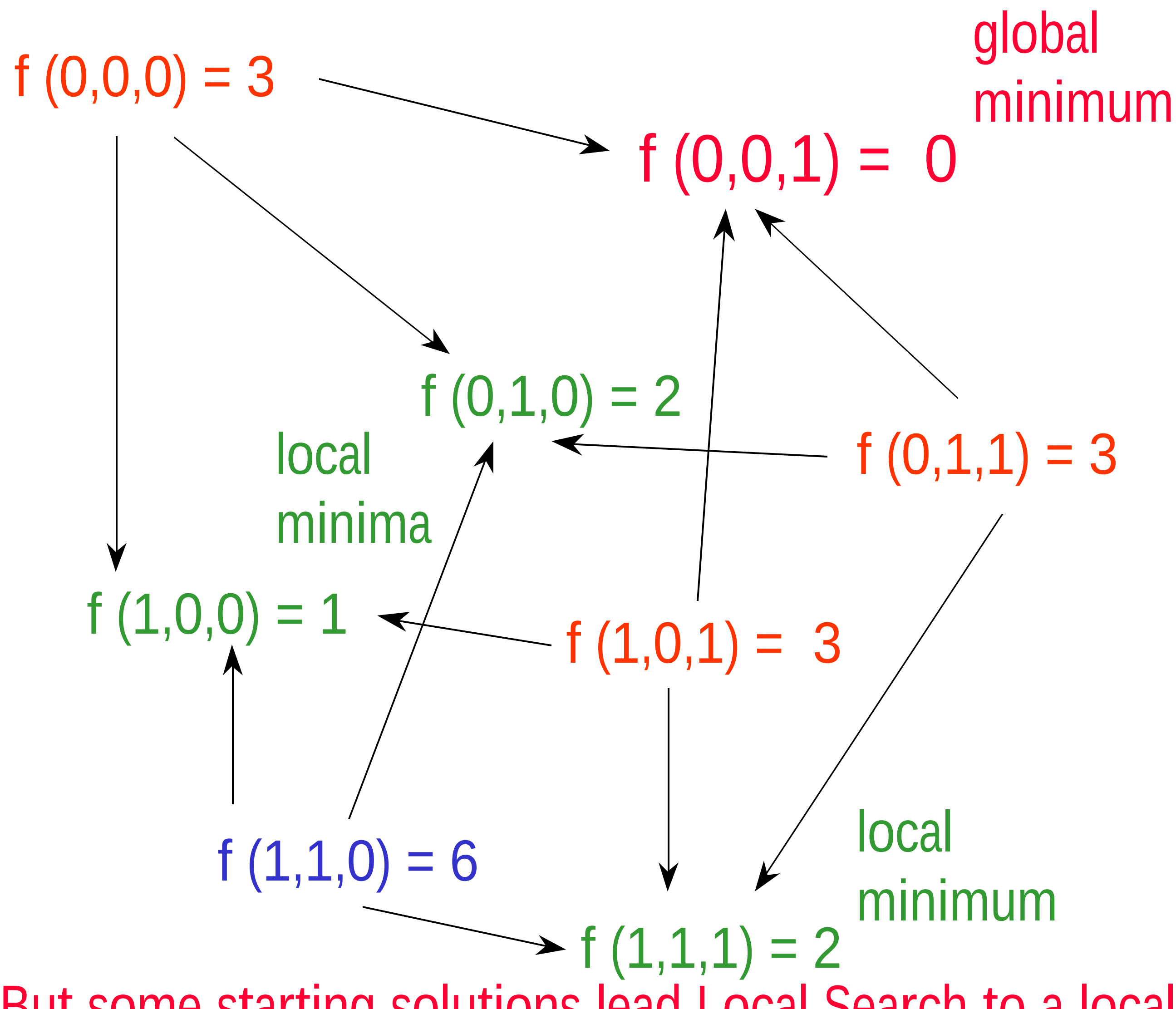
(ideal situation)



With any starting solution Local Search finds the global optimum.

Local Search

(more realistic situation)



But some starting solutions lead Local Search to a local minimum.

Local Search

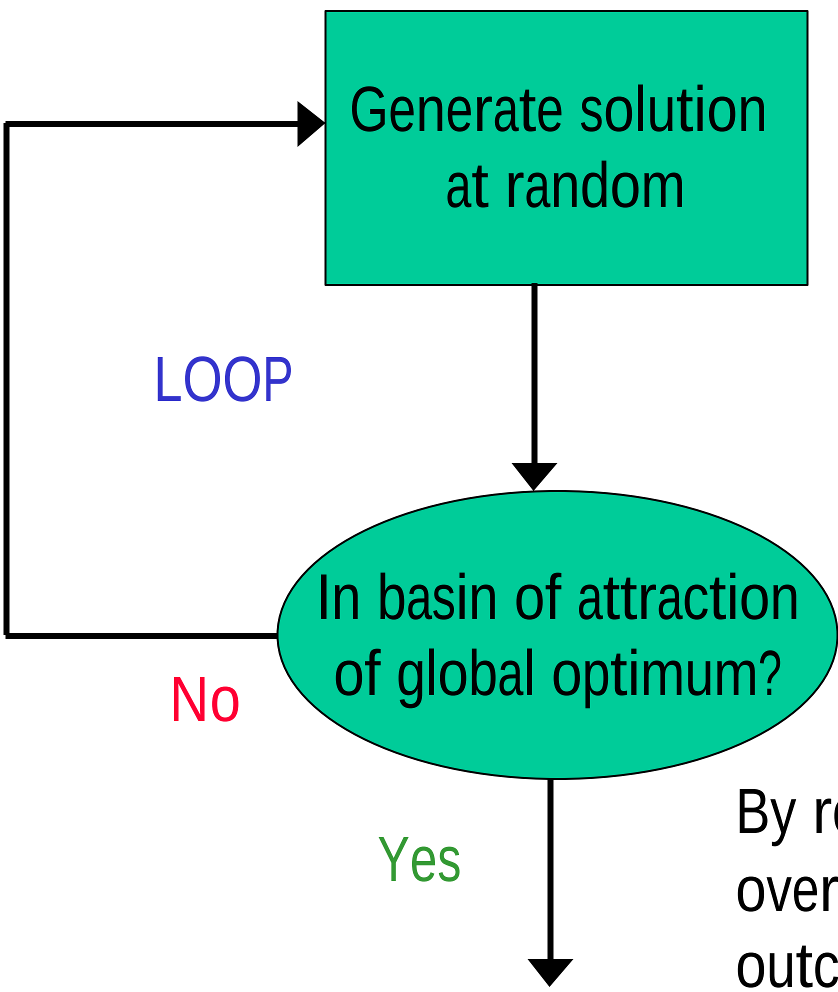
- Effectiveness of local search depends on several factors:
  - neighborhood structure
  - function to be minimized
  - starting solution

usually pre-determined (neighborhood structure, function to be minimized)

usually easier to control (starting solution)

AT&T

Local search with random starting solutions



By repeating LOOP over and over, w.p. 1 outcome is Yes

Local search leads to global optimum.

Random Multistart Local Search

```
best_obj = HUGE;  /* minimization */
repeat many times{
   x = random_construction( );
   x = local_search(x);
   if ( obj_function(x) < best_obj ){
         x* = x;
         best_obj = obj_function(x);
   }
}
```

The greedy algorithm

- To define a semi-greedy heuristic, we must first consider the greedy algorithm.
- Greedy algorithm: constructs a solution, one element at a time:

repeat until done

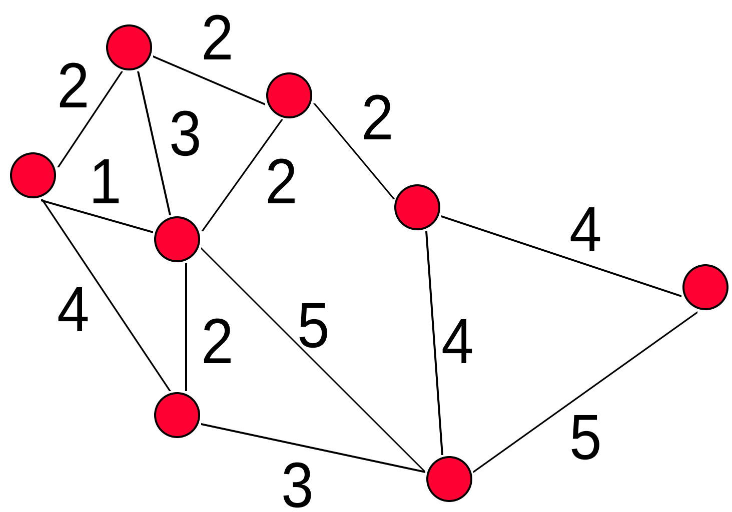
  - Defines candidate elements.
  - Applies a greedy function to each candidate element.
  - Ranks elements according to greedy function value.
  - Add best ranked element to solution.

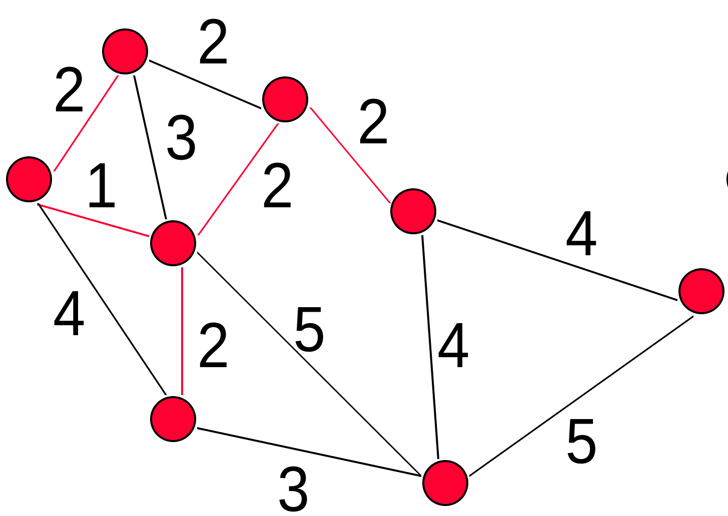
The greedy algorithm
An example

- Minimum spanning tree: Given graph G = (V, E), with weighted edges, find least weighted spanning tree.
  - greedy algorithm builds solution, one element (edge) at a time
  - greedy function: edge weight of edges that do not form cycle when added to current solution

The greedy algorithm

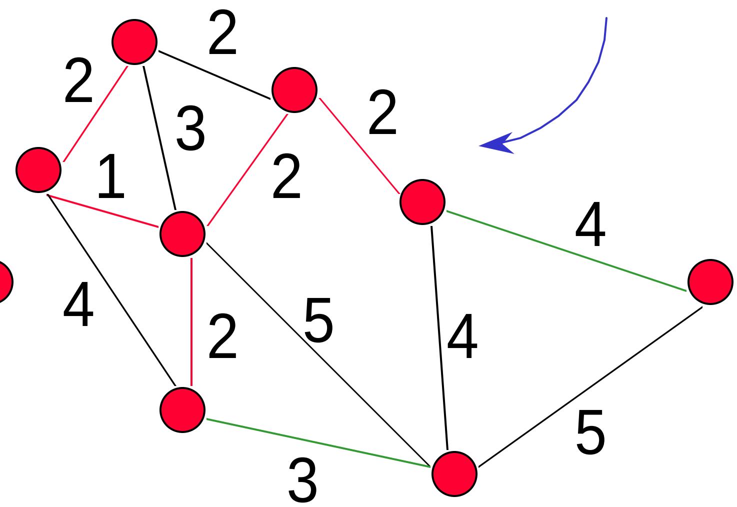
An example





Edges of weight 1 & 2

Global minimum

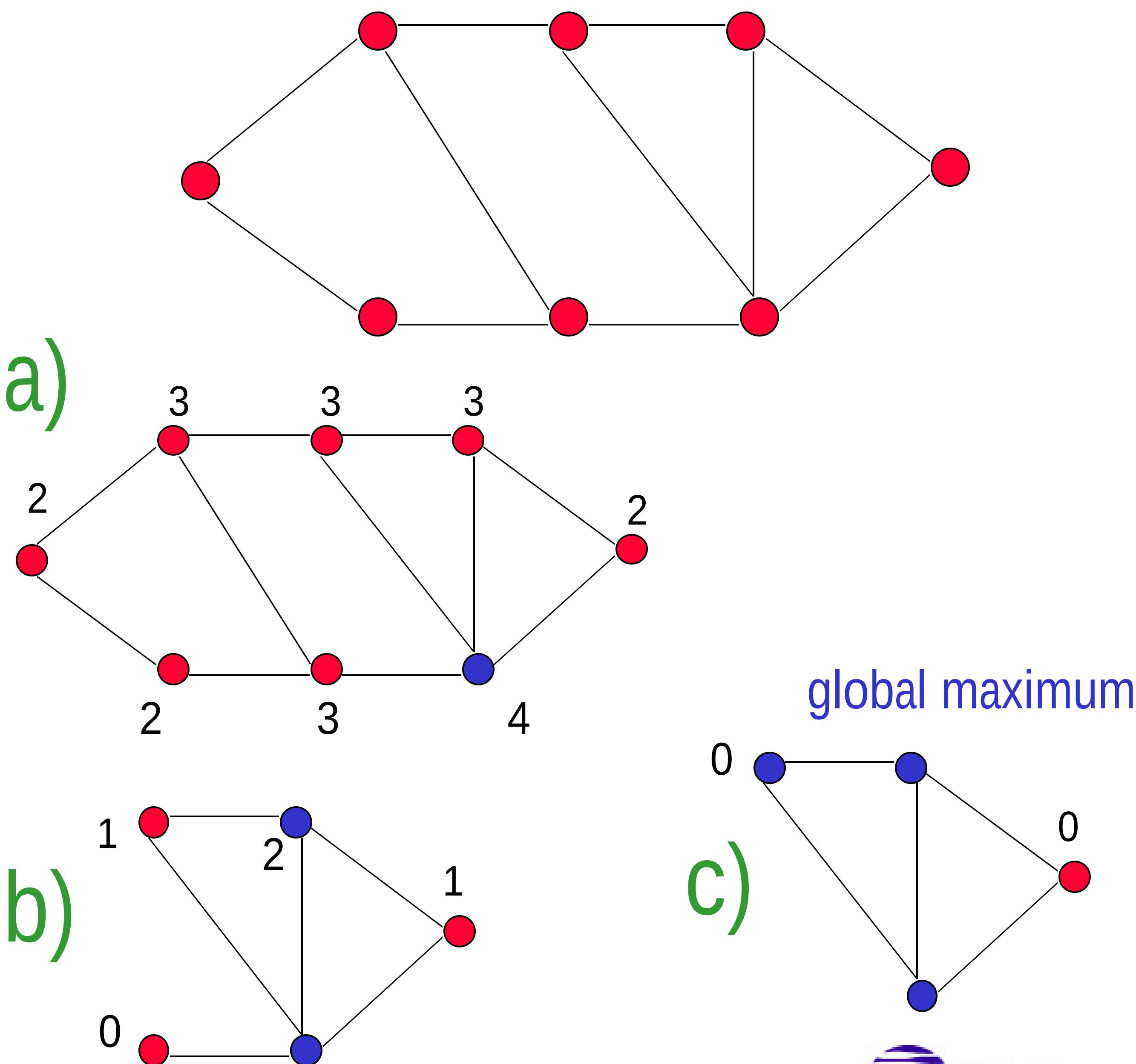


Edges of weight 3 & 4

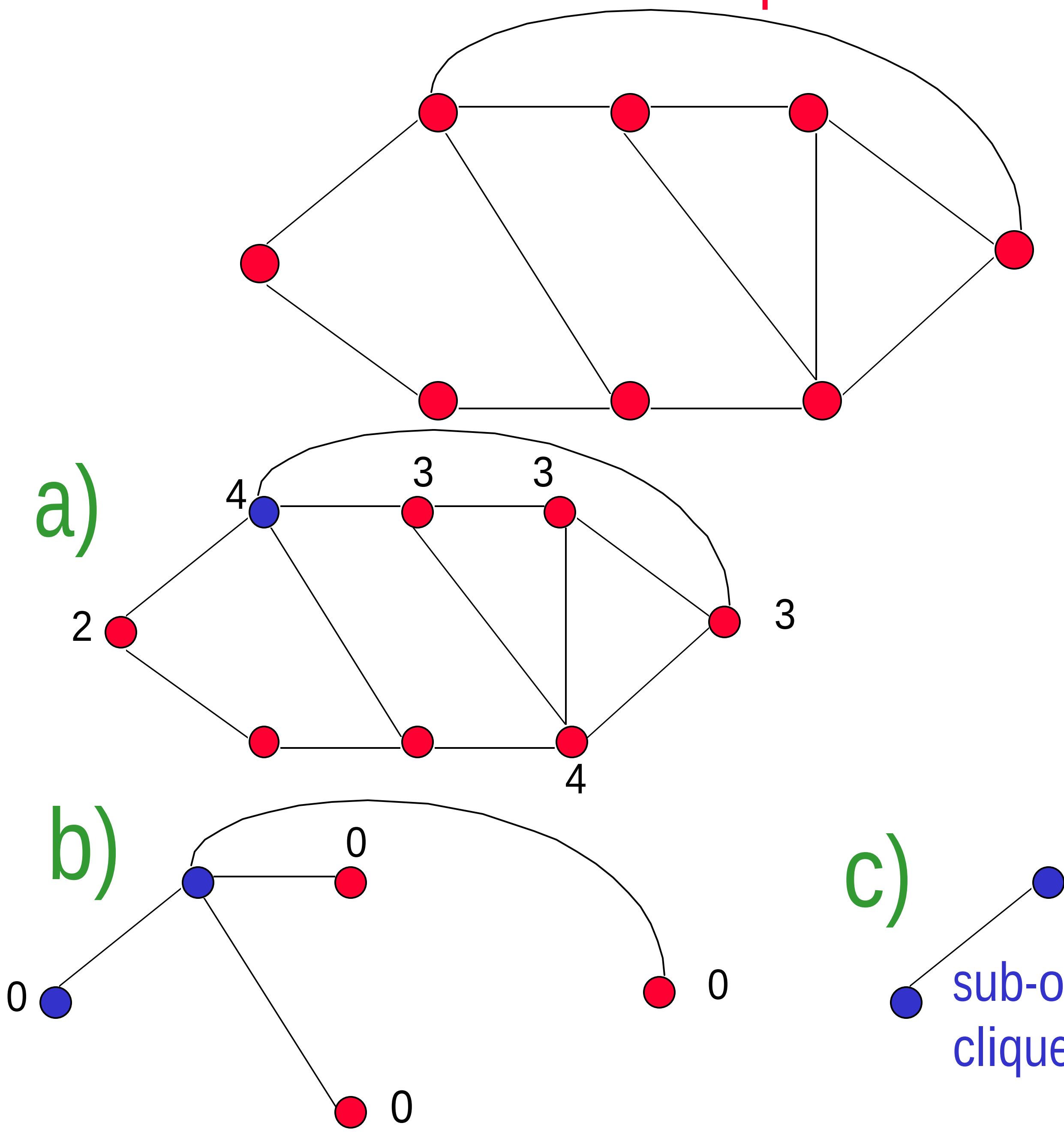
The greedy algorithm
Another example

- Maximum clique: Given graph G = (V, E), find largest subgraph of G such that all vertices are mutually adjacent.
  - greedy algorithm builds solution, one element (vertex) at a time
  - greedy function: degree of unselected vertex that is adjacent to all selected vertices with respect to all unselected vertices adjacent to all selected vertices.

The greedy algorithm
Another example



The greedy algorithm
Another example



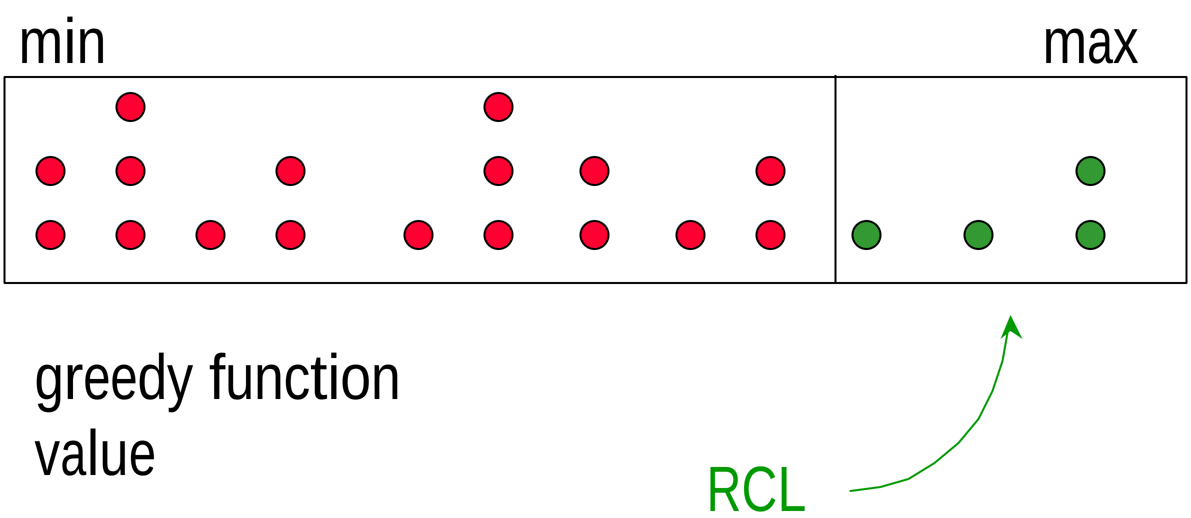
sub-optimal clique

Semi-greedy heuristic

- A semi-greedy heuristic tries to get around convergence to non-global local minima.
- repeat until solution is constructed
  - For each candidate element
    - apply a greedy function to element
  - Rank all elements according to their greedy function values
  - Place well-ranked elements in a restricted candidate list (RCL)
  - Select an element from the RCL at random & add it to the solution

Semi-greedy heuristic

Candidate elements are ranked according to greedy function value.



RCL is a set of well-ranked candidate elements.

Semi-greedy heuristic

- Hart & Shogan (1987) propose two mechanisms for building the RCL:
  - Cardinality based: place *k* best candidates in RCL
  - Value based: place all candidates having greedy values better than α*best_value in RCL, where $\alpha \in [0,1]$.
- Feo & R. (1989) proposed semi-greedy construction, independently, as a basic component of GRASP.

Hart-Shogan Algorithm
(maximization)

```
best_obj = 0;
repeat many times{
   x = semi-greedy_construction( );
   if ( obj_function(x) > best_obj ){
         x* = x;
         best_obj = obj_function(x);
   }
}
```

A Basic GRASP

- GRASP tries to capture good features of greedy & random constructions.
- iteratively
  - samples solution space using a greedy probabilistic bias to construct a feasible solution (semi-greedy construction)
  - applies local search to attempt to improve upon the constructed solution
- keeps track of the best solution found

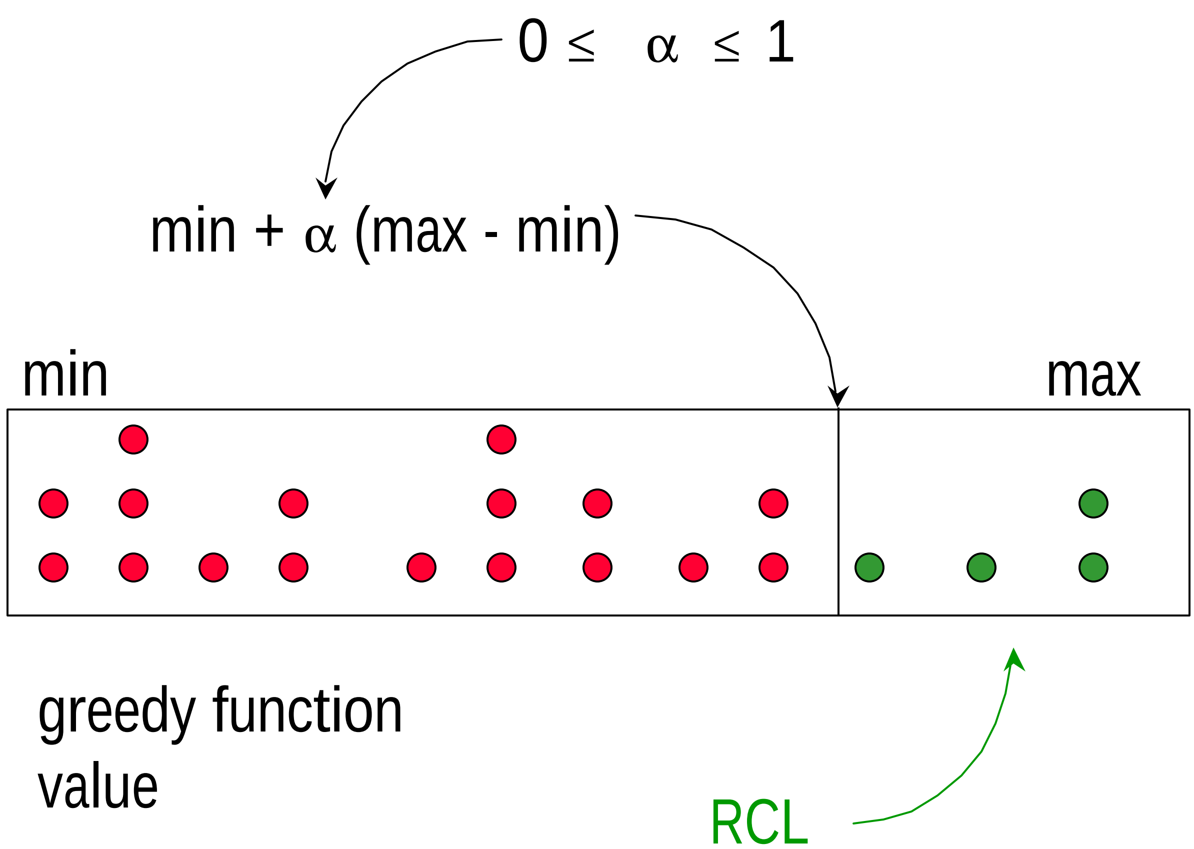
GRASP
(for maximization)

```
best_obj = 0;
repeat many times{
    x = semi-greedy_construction( );
    x = local_search(x);
    if ( obj_function(x) > best_obj ){
        x* = x;
        best_obj = obj_function(x);
    }
}
```

bias towards greediness

good diverse solutions

minmax α - percentage based RCL



α = 0: random assignment

α = 1: greedy assignment

Random vs greedy construction

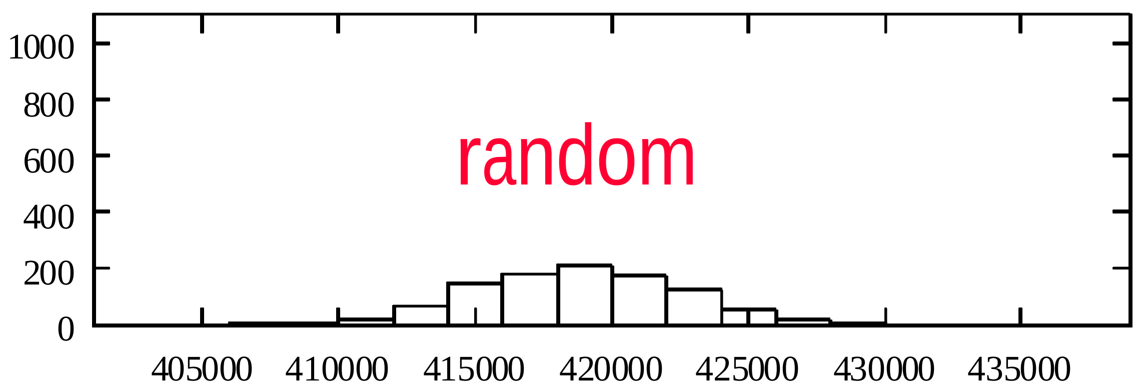
$\alpha = 0$

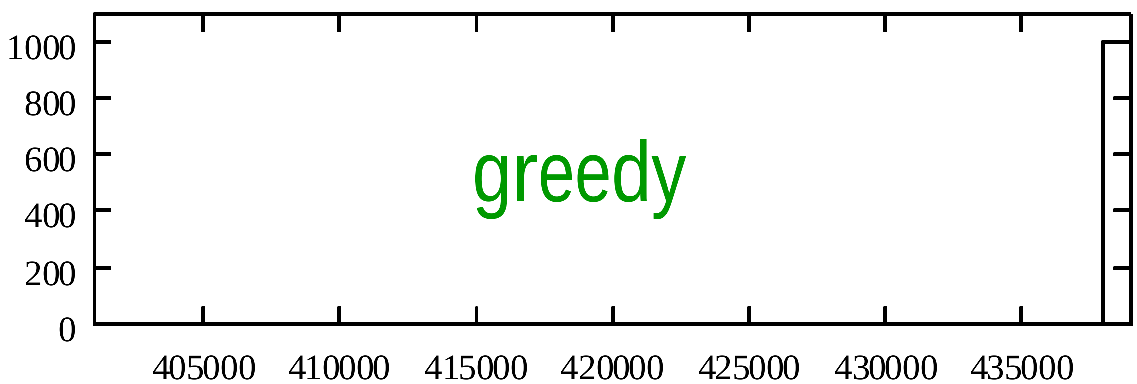
- random construction
  - high variance
  - low quality
  - almost always sub-optimal

$\alpha = 1$

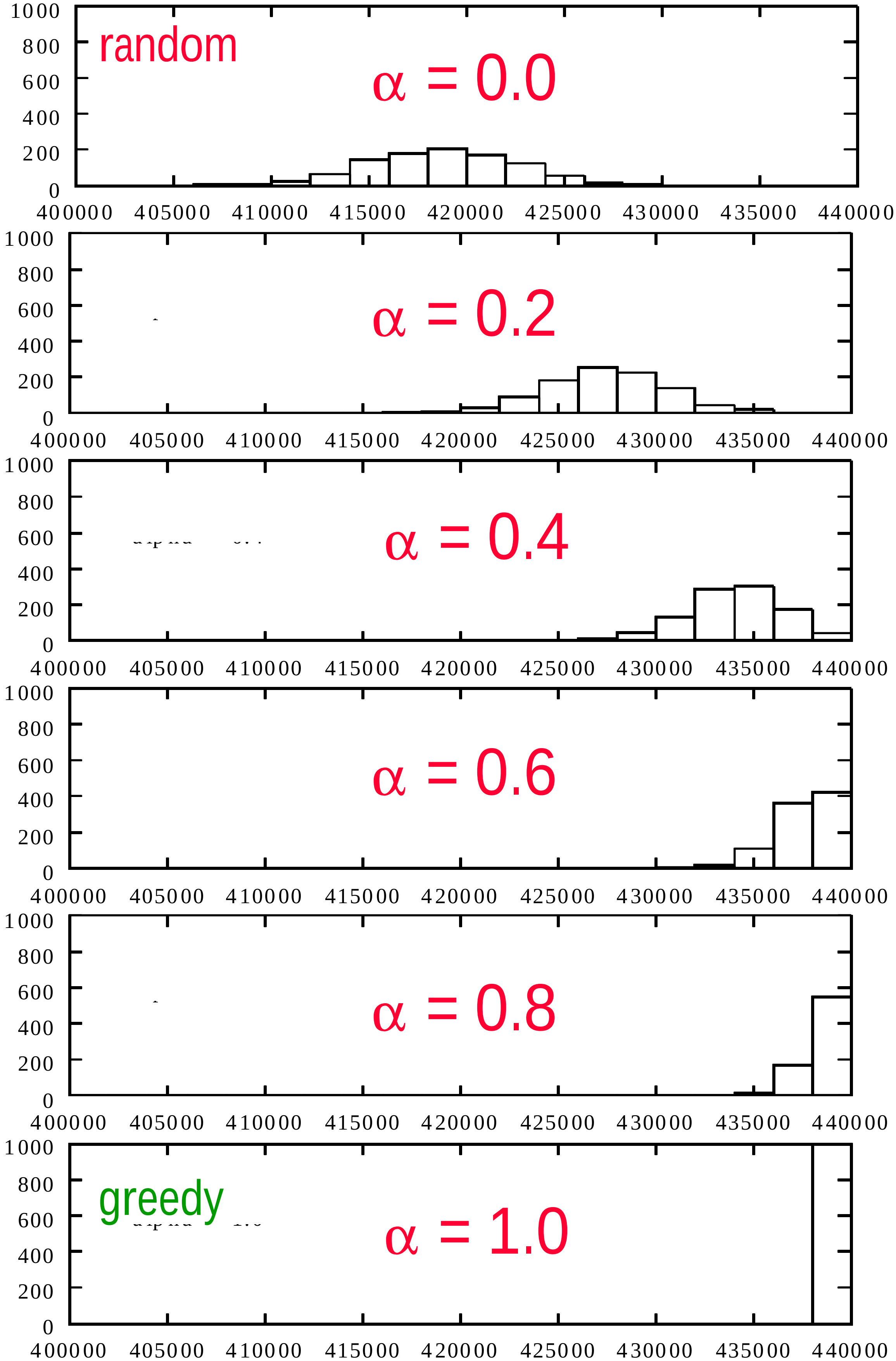
- greedy construction
  - low (no) variance
  - good quality
  - usually sub-optimal

Random vs greedy construction





Construction phase solutions



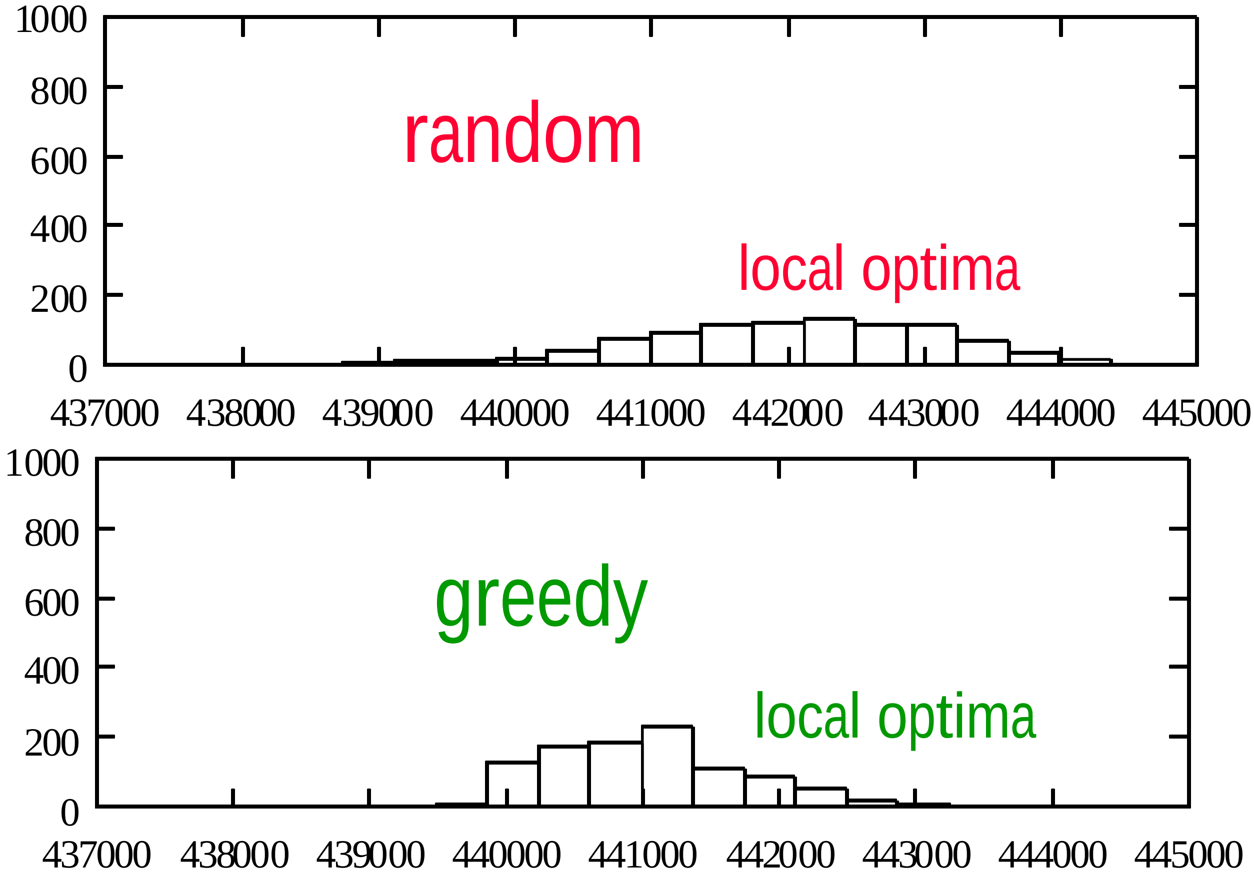
How do methods compare?

- Local search from random starting solution:
  - high variance
  - avg solution worse than avg greedy
  - best solution usually better than best greedy
  - slow convergence

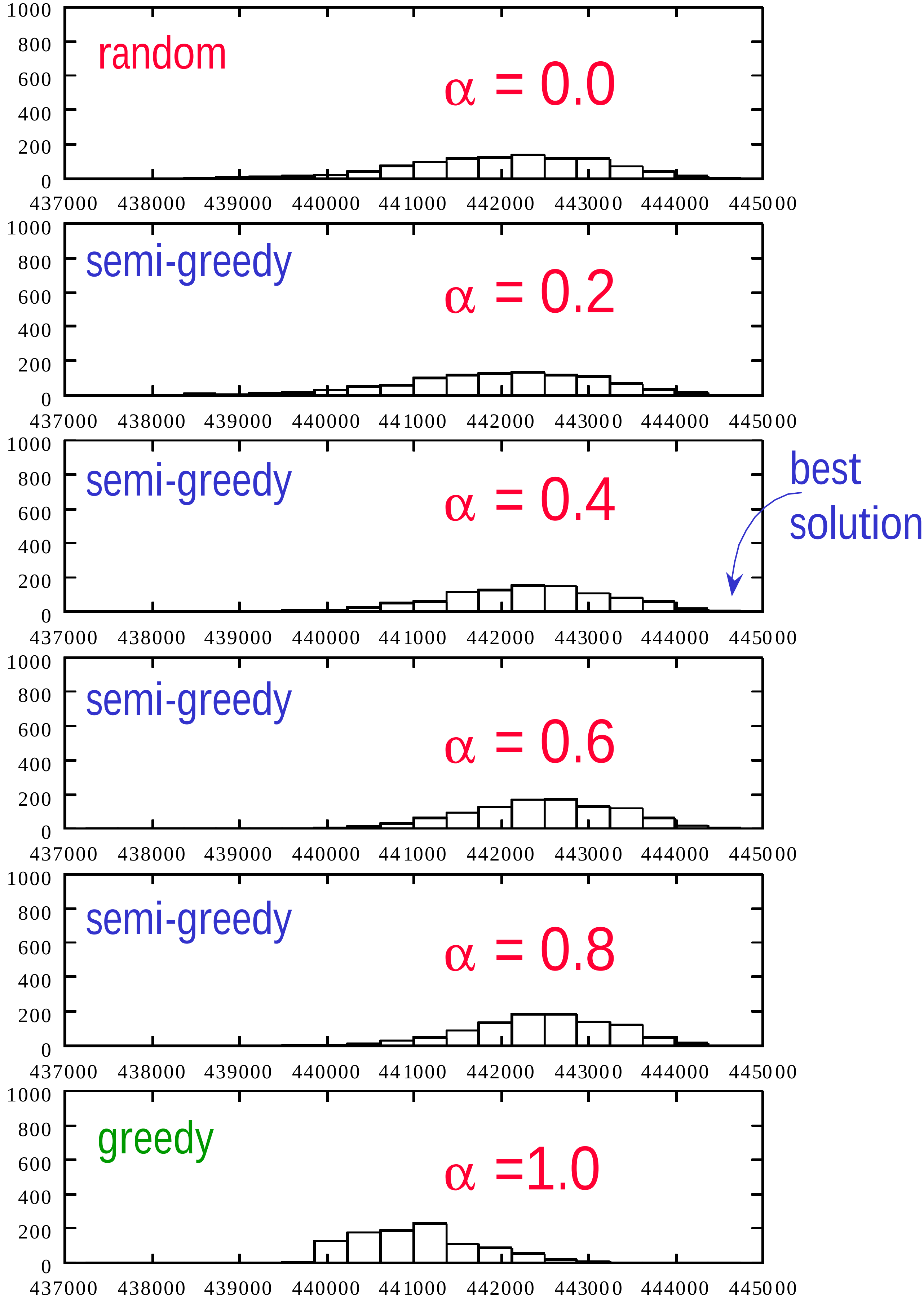
- Local search from greedy starting solution:
  - low (no) variance
  - usually sub-optimal
  - fast convergence

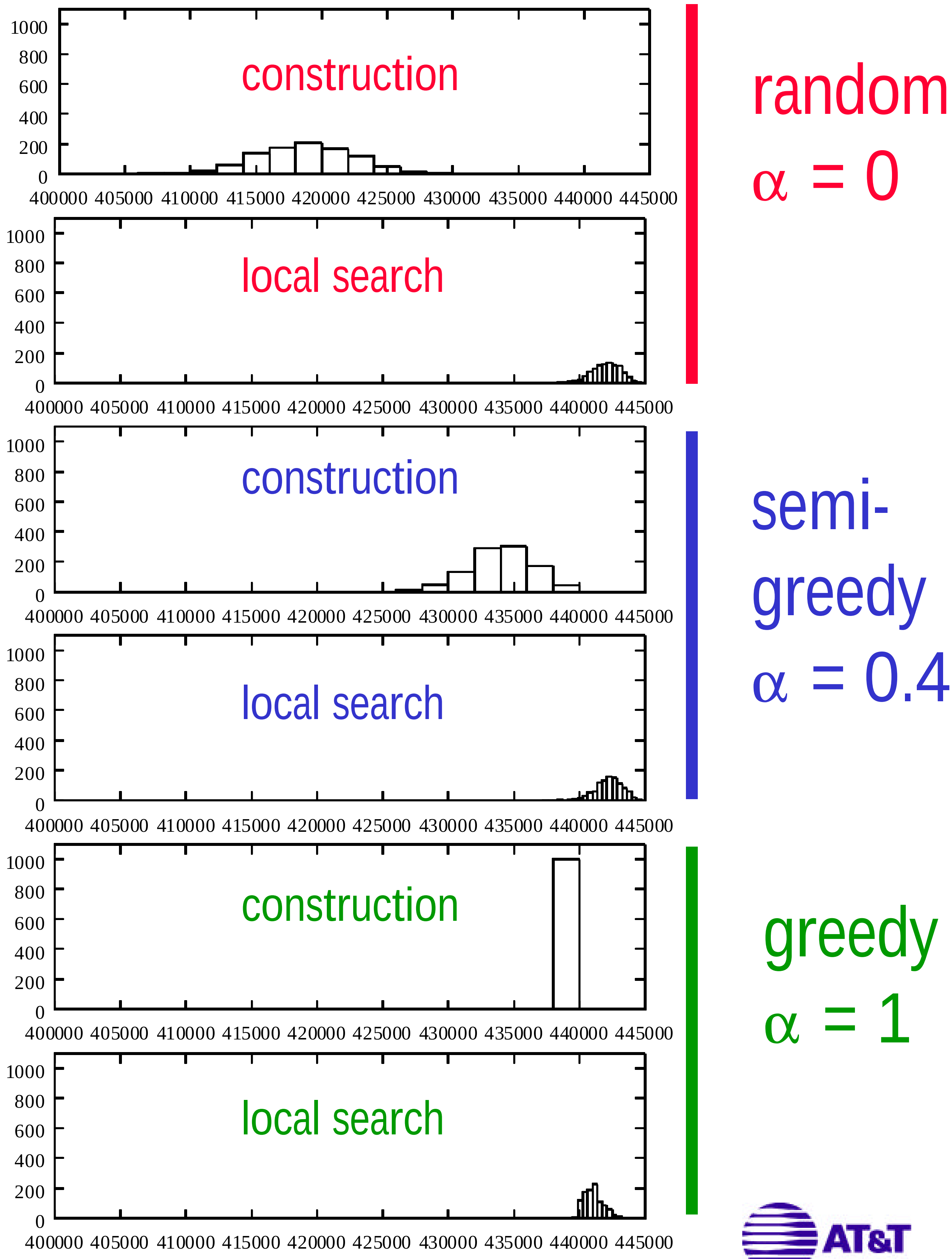
GRASP tries to capture good features of greedy & random constructions.

Greedy vs Random: As starting solution for local search



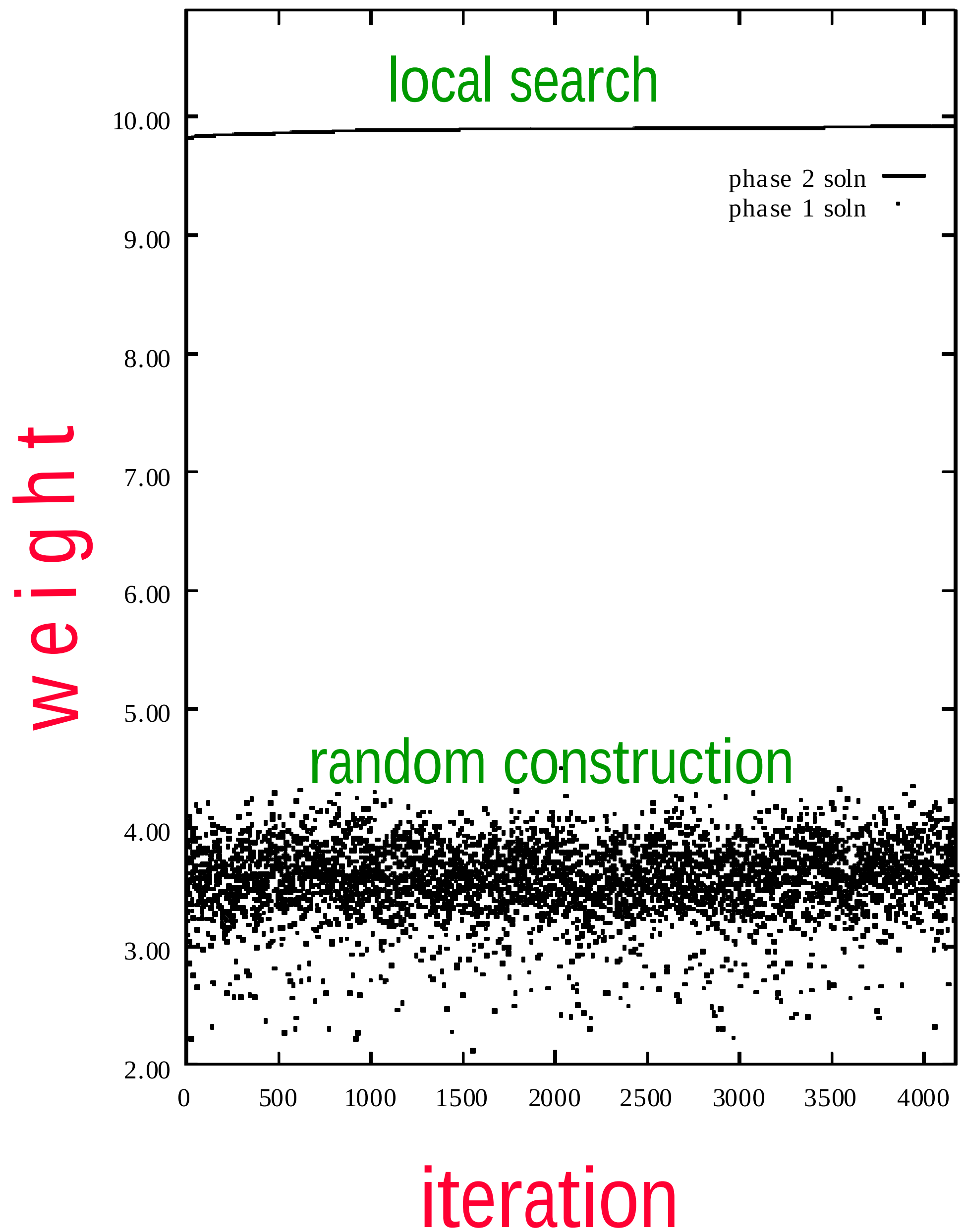
Local search phase solutions



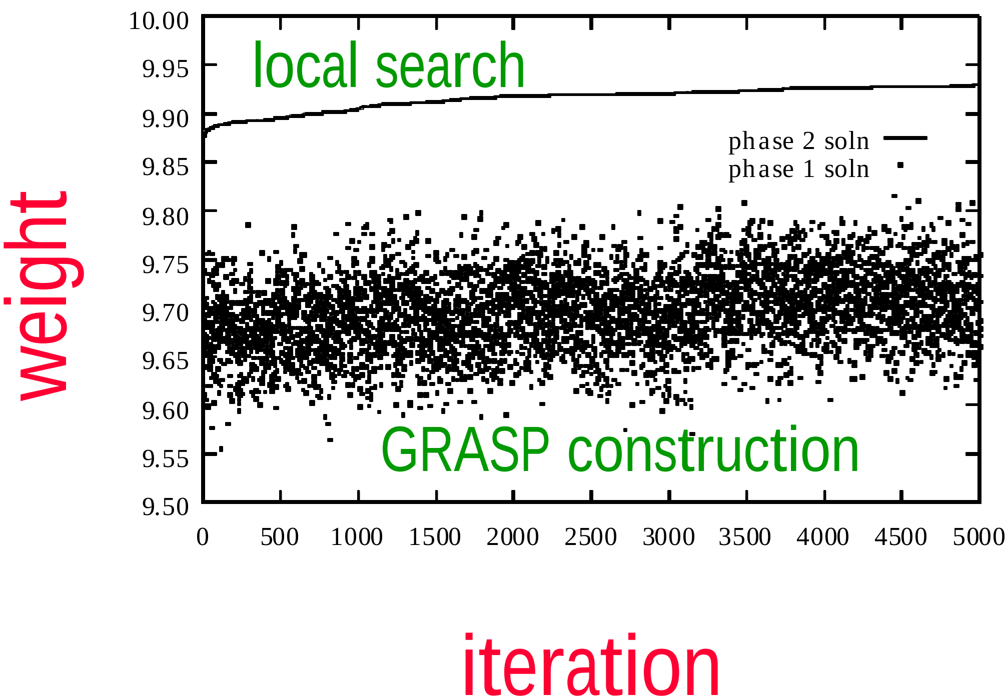

construction
local search
construction
local search
construction
local search
1000
800
600
400
200
0
400000 405000 410000 415000 420000 425000 430000 435000 440000 445000
random
α = 0
semi-
greedy
α = 0.4
greedy
α = 1

AT&T

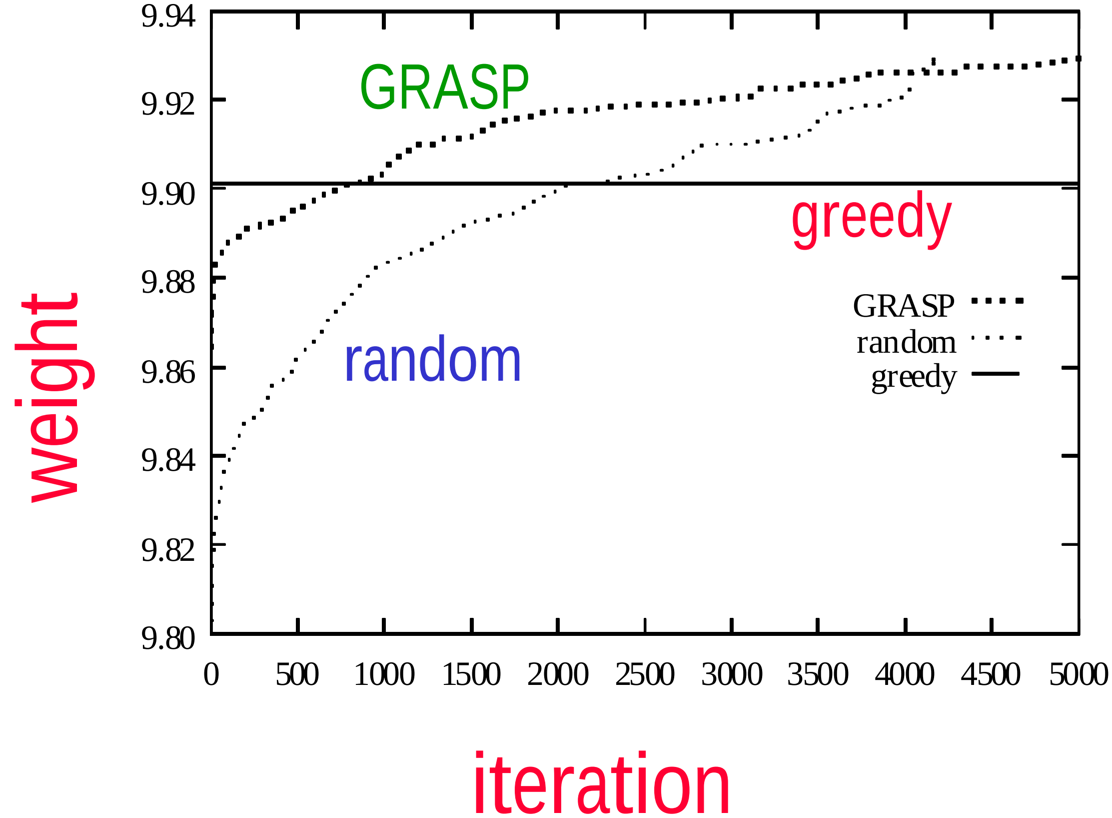
Random & local search



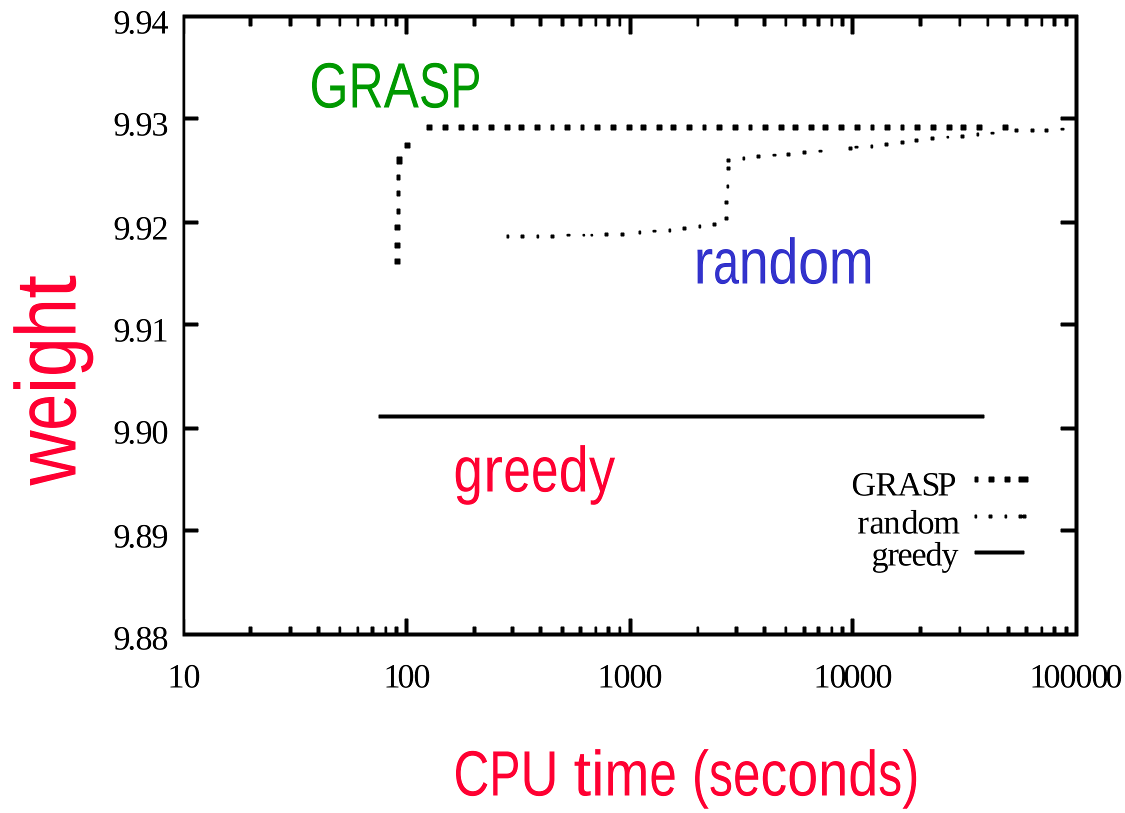
GRASP ($\alpha = 0.85$)



Local search solutions



Local search solutions



Local search

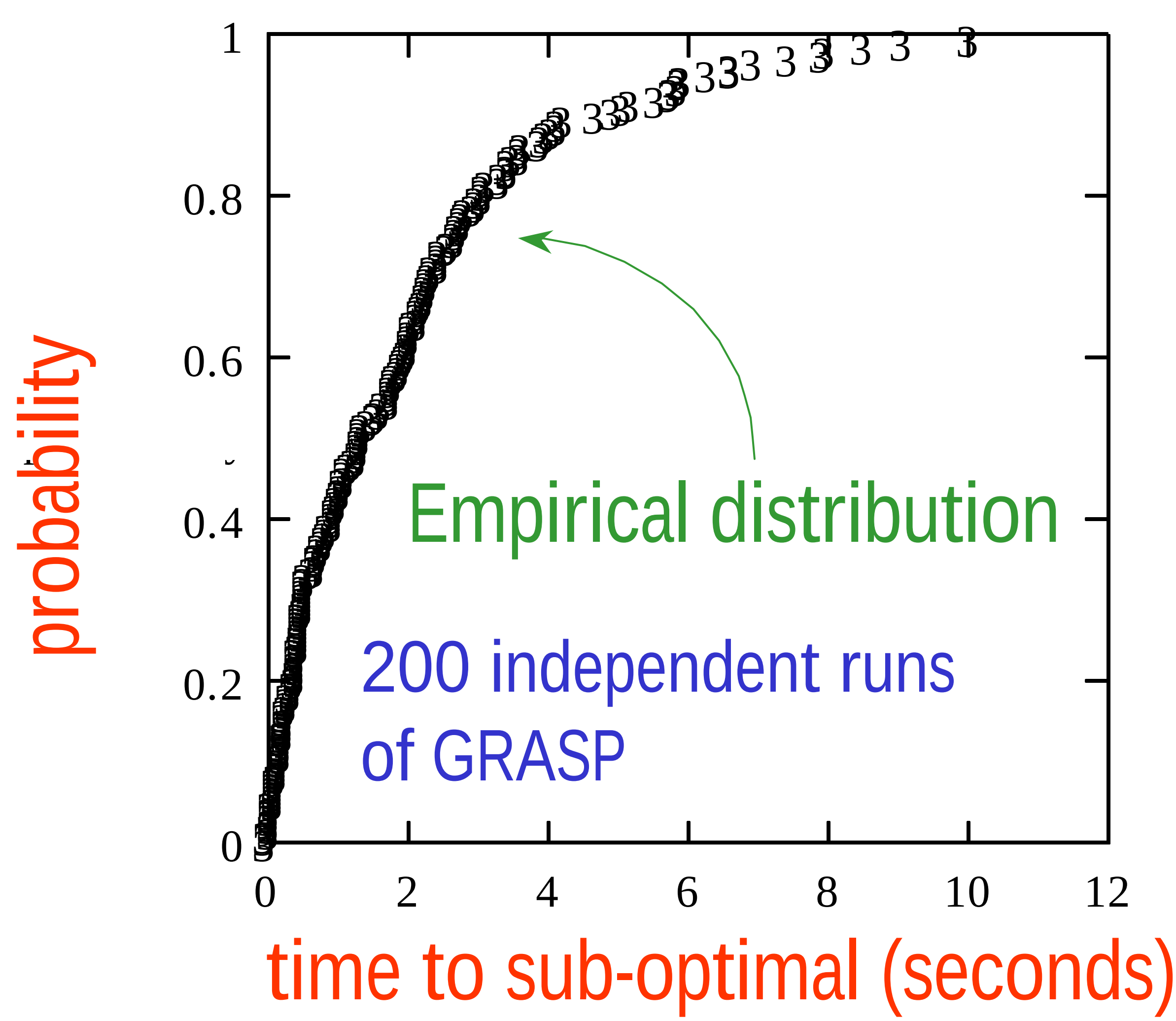
- Local search is done from constructed solution:
  - to improve constructed solution that is not locally optimal
  - to improve constructed solution that is locally optimal
- Types of local search used:
  - exchange [e.g. Feo, R., & Smith, 1994; Laguna, Feo, Elrod, 1994]
  - tabu search [Laguna & Velarde, 1991; Díaz & Fernández, 1998]
  - simulated annealing [Feo & Smith, 1994]
  - path relinking [Laguna & Martí, 1999]
  - POP [Fleurent & Glover, 1999]

AT&T

Probability distribution: Time to sub-optimal

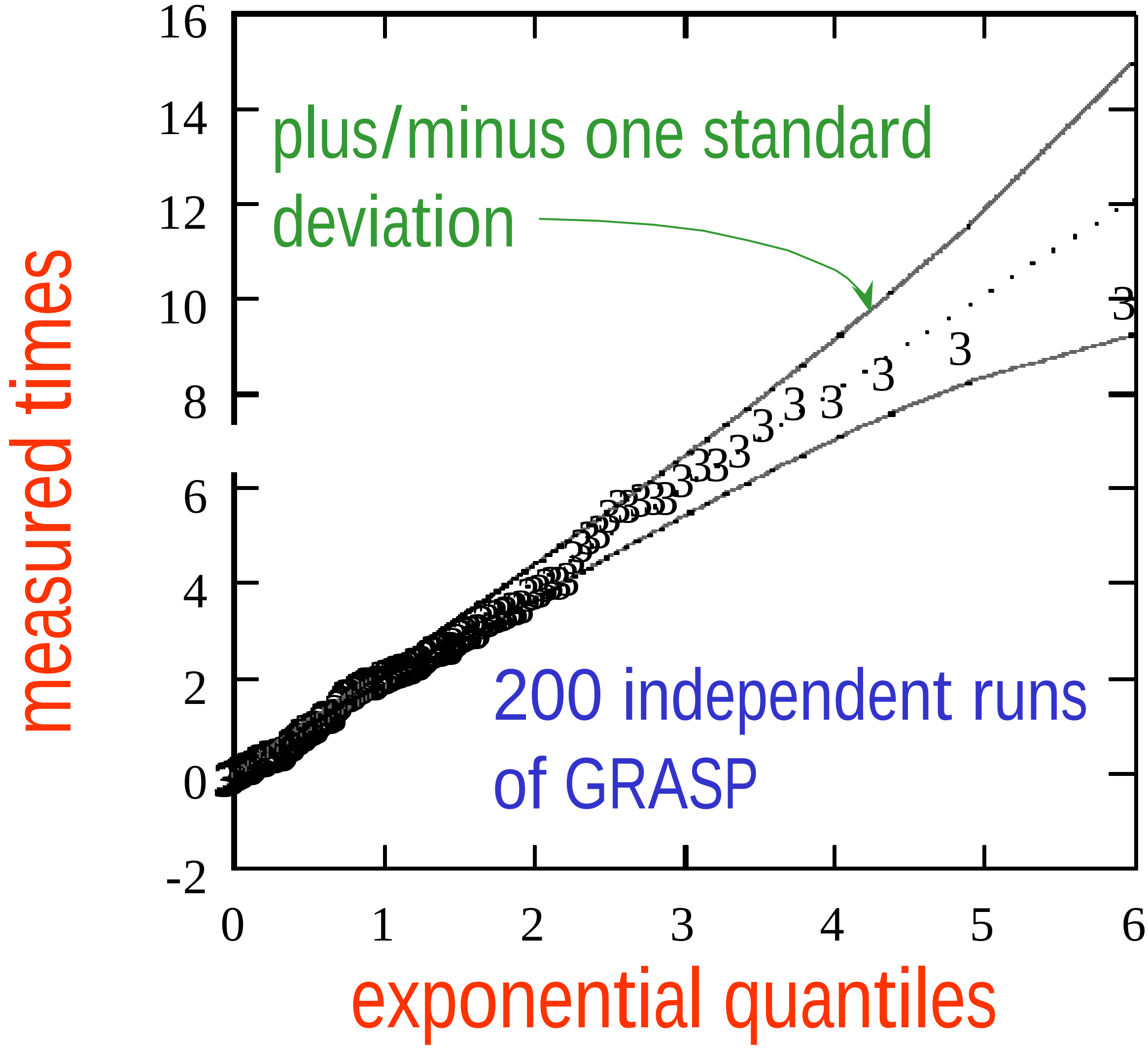
- Aiex, R., & Ribeiro (2000) studied the probability distribution of "time to sub-optimal" of several GRASPs
  - showed that "time to sub-optimal" fits a two-parameter (or shifted) exponential distribution
  - this has important implications regarding parallel implementations of GRASP

Probability distribution: Time to sub-optimal

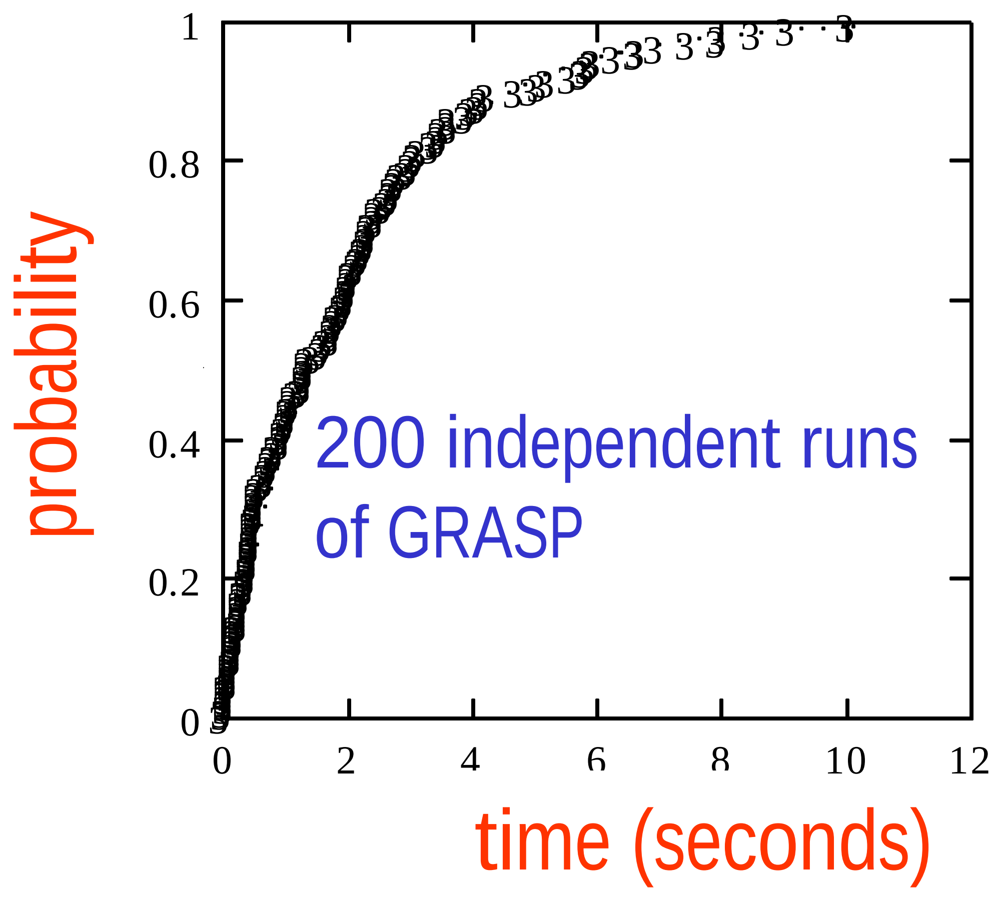


AT&T

Q-Q plot with variability information



Superimposed empirical & theoretical distributions

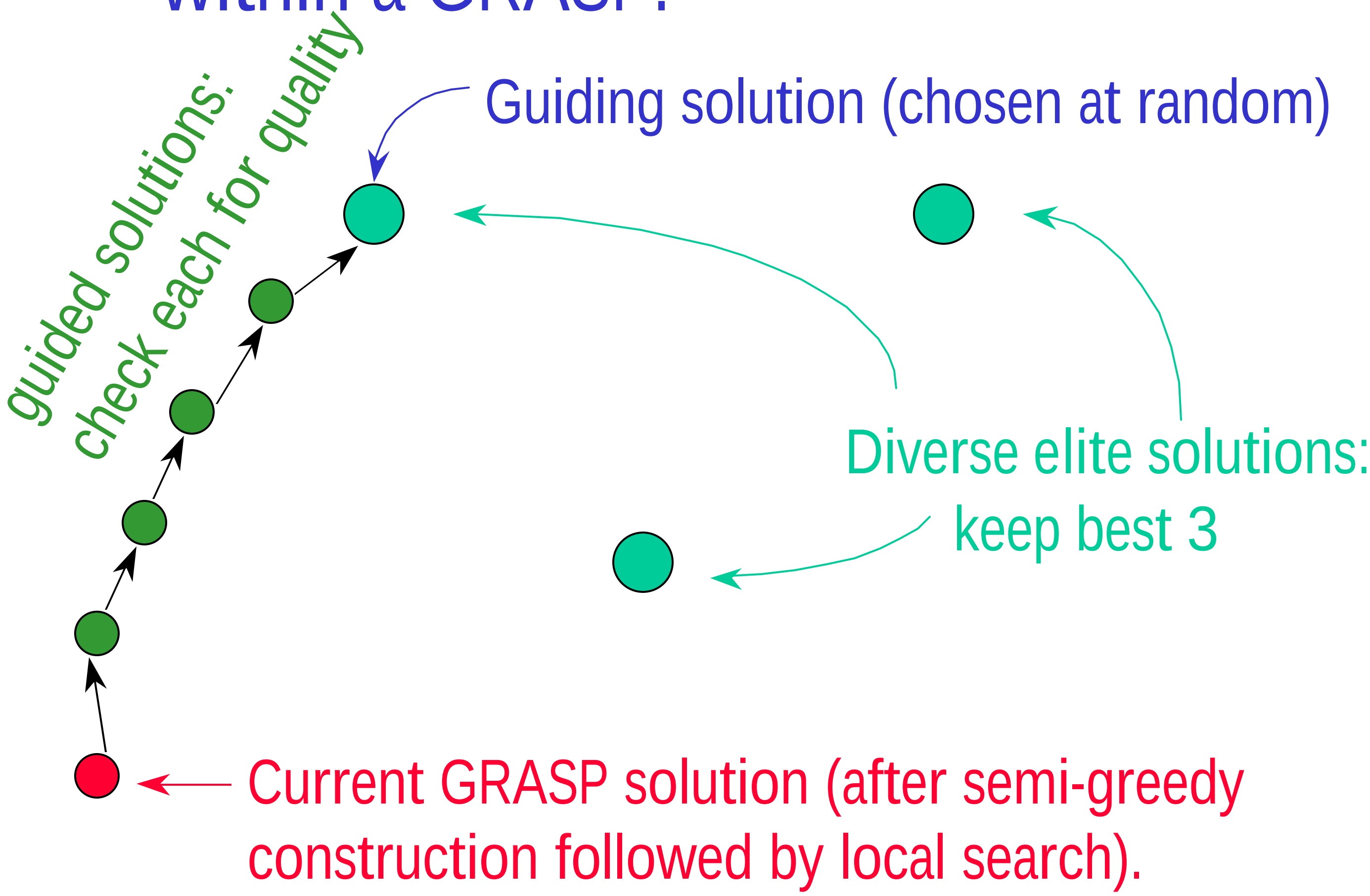


Enhancements

- Local search
  - Path relinking
  - Proximate Optimality Principle (POP)
- Asymptotically convergent GRASP
  - Bias function
- Automatic choice of RCL parameter α
  - Reactive GRASP
- Use of long-term memory
- GRASP and Genetic Algorithms
- Parallel GRASP

Path relinking

- Laguna & Martí (1999) adapted the concept of path relinking for use within a GRASP.



Path relinking

X is current GRASP iterate

Y is guiding solution from elite set

Δ = symmetric difference (X,Y)

Example:

X = (1,1,0,1,0)

Y = (1,0,0,0,1)

$\Delta = (*, 1 \rightarrow 0, *, 1 \rightarrow 0, 0 \rightarrow 1)$

while (Δ is not empty){

evaluate each move in Δ from X

let δ be the best move

X = move (X, δ)

if (X is better than X^*) { X^* = X }

$\Delta = \Delta \setminus \{ \delta \}$

}

Best solution is tested for membership In Elite set after P.R.

Path relinking

- Aiex, Pardalos, R., & Toraldo (2000) and Festa, R., & Pardalos (2000) added the following to the approach of Laguna & Martí:
  - Large elite sets (10 to 50 elements)
  - Back and forth path relinking
  - Path relinking between solution and all elite solutions
  - Test for inclusion into elite set only best solution in path
  - Intermediate and post-optimization path relinking between all elite set solutions

Proximate Optimality Principle (POP)

- "Good solutions at one level are likely to be found 'close to' good solutions at an adjacent level." [Glover & Laguna, 1997]
- GRASP interpretation of POP: imperfections introduced during steps of GRASP construction can be "ironed-out" by applying local search during (and not only at the end of) GRASP construction [Fleurent & Glover, 1999].

Proximate Optimality Principle (POP)

- POP has been applied in GRASPs for:
  - QAP by Fleurent & Glover (1999)
  - Job shop scheduling by Binato, Hery, Loewenstern, and R. (1999)
  - transmission expansion planning by Binato & Oliveira (1999)
- In all instances, POP improved the performance of GRASP.

Convergent GRASP

- Mockus, Eddy, Mockus, Mockus, & Reklaitis (1997) pointed out that GRASP with fixed nonzero RCL parameter α may not converge (asymptotically) to a global optimum.
- Remedies:
  - Randomly select α uniformly from the interval [0,1] [R., Pitsoulis, & Pardalos, 1998]
  - Use bias function selection mechanism of Bresina [1996]
  - Reactive GRASP [Prais & Ribeiro, 1998]

Bias function

- Bresina (1996) introduced the concept of a bias function to select a candidate element to be included in the solution.
  - rank all candidate elements by greedy function values
  - assign bias(r) to r-th ranked element
    - logarithmic: bias(r) = $1/\log(r+1)$
    - linear: bias(r) = $1/r$
    - polynomial(n): bias(r) = $1/r^n$
    - exponential: bias(r) = $1/e^r$
    - random: bias(r) = 1

Bias function

- define total_bias = Σ bias(r)
- assign probability of selection of the element ranked r to be:
  prob(r) = bias(r) / total_bias
- pick r -th ranked element with probability prob(r)

- Binato, Hery, Loewenstern, & R. (2000) use bias function to select an element from the RCL.

Automatic choice of RCL parameter α

- Choice of RCL parameter α is complicated:
  - may be problem dependent
  - may be instance dependent
- Remedies:
  - Randomly selected RCL parameter [R., Pitsoulis, & Pardalos, 1998].
  - Reactive GRASP [Prais & Ribeiro, 1998]: self-adjusting α according to previously found solutions.

Reactive GRASP

- Introduced by Prais & Ribeiro (1998)
- At each GRASP iteration, a value of the RCL parameter α is chosen from a discrete set of values $\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$
- The probability that α_k is selected is $p(\alpha_k)$.
- Reactive GRASP adaptively changes the probabilities $\{p(\alpha_1), p(\alpha_2), \ldots, p(\alpha_m)\}$ to favor values that produce good solutions.

Reactive GRASP

- We describe Reactive GRASP for a minimization problem.
- Initially $p(\alpha_i) = 1/m$, $i = 1, 2, \ldots, m$, i.e. values are selected uniformly.
- Define
  - $F(S^*)$ be the value of the incumbent (i.e. best so far) solution.
  - A_i be the average value of the solutions obtained with α_i in the construction phase.

Reactive GRASP

- Compute every N_α iterations:
  - $q_i = \left(F(S^*) / A_i \right)^\delta, \; i = 1,2, \ldots, m$
  - $p(\alpha_i) = q_i / \Sigma q_j, \; i = 1,2, \ldots, m$
- Observe that the more suitable a value α_i is, the larger the value of q_i is and, consequently, the higher the value of $p(\alpha_i)$, making α_i more likely to be selected.
- The parameter δ can be used as an attenuation parameter.

AT&T

Reactive GRASP

- Has been applied to:
  - traffic scheduling in satellite switched time division multi-access (SS/TDMA) systems [Prais & Ribeiro, 1998]
  - single source capacitated plant location [Díaz & Fernández, 1998]
  - transmission expansion planning [Binato & Oliveira, 1999]
  - mobile phone frequency assignment [Oliveira, Gomes, & R., 2000]
- Extensive computational experiments described in [Prais & Ribeiro, 1999]

Long term memory

- Since GRASP iterations are independent, current iteration makes no use of information gathered in previous iterations.
- Remedies:
  - Path relinking [Laguna & Martí, 1999]
  - Reactive GRASP [Prais & Ribeiro, 1998]
  - Use set of previously generated elite solutions to guide construction phase of GRASP [Fleurent & Glover, 1999] as an intensification mechanism.

Fleurent & Glover intensification scheme

- Introduced as a way to use long term memory in multi-start heuristics such as GRASP [Fleurent & Glover, 1999]
- An elite set of solutions *S* is maintained. To be in *S* solution must be:
  - better than best member of *S* , or
  - better than worst and sufficiently different from other elite solutions
    - e.g. count identical vector components and set a threshold for rejection
- Use elite set in construction phase.

Fleurent & Glover intensification scheme

- Strongly determined variables are those that cannot be changed without eroding the objective or changing significantly other variables.
- A consistent variable is one that receives a particular value in a large portion of the elite solution set.
- Let $I(e)$ be a measure of the strongly determined and consistent features of choice e, i.e. $I(e)$ becomes larger as e resembles solutions in elite set S

Fleurent & Glover intensification scheme

- Intensity function is used in the construction phase
  - Recall $g(e)$ is greedy function
  - Let $E(e) = F(g(e), I(e))$
    - e.g. $E(e) = \lambda\, g(e) + I(e)$
  - Bias selection from RCL to those elements with a high $E(e)$ value.
    - prob (selecting e) $= E(e) \,/\, \Sigma_{s\, \varepsilon\, RCL}\, E(s)$
- $E(e)$ can vary with time (e.g. changing the value of λ)
  - keep λ large initially, then reduce
  - to add diversification, increase λ

Fleurent & Glover intensification scheme

- Has been applied to
  - QAP [Fleurent & Glover, 1999]
  - Job shop scheduling [Binato, Hery, Loewenstern, & R., 1999]

GRASP in hybrid metaheuristics

- tabu search as local search phase [Laguna & González-Velarde, 1991; Colomé & Serra, 1998; Delmaire, Díaz, Fernández, & Ortega, 1999]
- simulated annealing as local search phase [Feo & Smith, 1994; Liu, Pardalos, Rajasekaran, & R., 2000]
- path relinking as additional local search phase [Laguna & Martí, 1999; Festa, Pardalos, & R., 2000; Aiex, Pardalos, R., & Toraldo, 2000]

GRASP in hybrid metaheuristics

- GRASP as initial population generator for genetic algorithms (GA) [Ahuja, Orlin, & Tiwari, 2000]

- GRASP has also been used in a GA to implement a crossover operator that generates perfect offspring [Ramalhinho, Paixão, & Portugal, 1998]
  - Given two parents, perfect offspring are the best possible offspring and their determinations requires the solution of an optimization problem.

Parallel GRASP

- GRASP is easy to implement in parallel:
  - parallelization by problem decomposition
    - Feo, R., & Smith (1994)
  - iteration parallelization
    - Pardalos, Pitsoulis, & R. (1995)
    - Pardalos, Pitsoulis, & R. (1996)
    - Alvim (1998)
    - Martins & Ribeiro (1998)
    - Murphey, Pardalos, & Pitsoulis (1998)
    - R. (1998)
    - Martins, R., & Ribeiro (1999)
    - Aiex, Pardalos, R., & Toraldo (2000)

Parallel GRASP

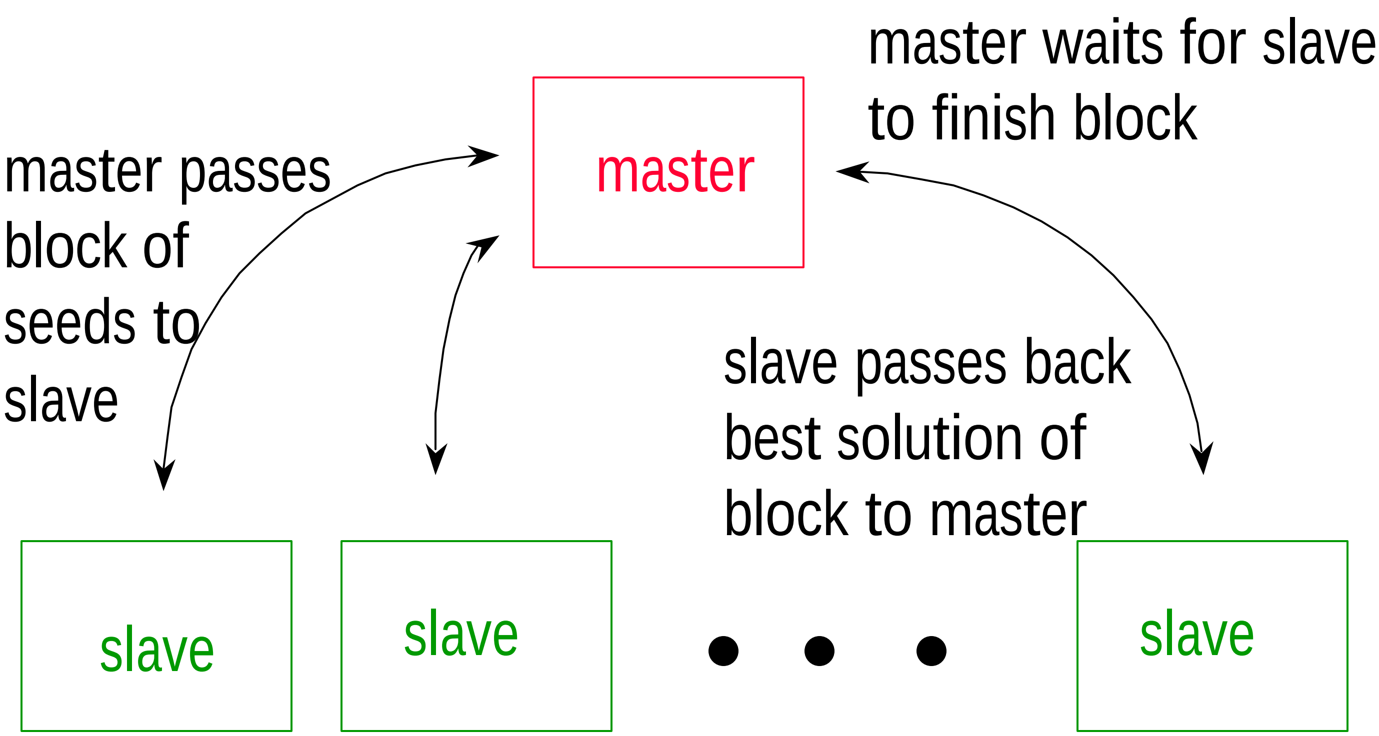
- Let $P_r(t)$ be the probability of not having found a given (target) solution in t time units with r independent processes.
  - If $P_1(t) = \lambda\, e^{-(t-\mu)/\lambda}$, where λ and μ are real numbers ($\lambda > 0$),
  - Then $P_r(t) = r\, \lambda\, e^{-(t-\mu)/(\mu\lambda)}$
- Probability of finding a target solution in time rt with one processor equals probability of finding a solution at least as good in time t with r processors.
- Experiments indicate that this is the case for memoryless implementations of GRASP [Aiex, R. & Ribeiro, 2000].

Simple parallelization

- Most straightforward scheme for parallel GRASP is distribution of iterations to different processors.
- Care is required so that two iterations never start off with same random number generator seed.
  - run generator and record all N_g seeds in seed( ) array
  - start iteration *i* with seed seed(*i*)

PVM & MPI implementations

- PVM: Pardalos, Pitsoulis, & R. (1996)
- MPI: Alvim (1998); Alvim & Ribeiro (1998); Martins, R., & Ribeiro (1999); Aiex, Pardalos, R., & Toraldo (2000)



Survey of O.R. & C.S. applications in literature

- scheduling
- routing
- logic
- partitioning
- location
- graph theoretic
- QAP and other assignment problems
- miscellaneous problems

Scheduling

- operations sequencing [Bard & Feo, 1989]
- flight scheduling [Feo & Bard, 1989]
- single machine [Feo, Venkatraman, & Bard, 1991]
- just-in-time scheduling [Laguna & González-Velarde, 1991]
- Constant flow allowance (CON) due date assignment & sequencing [De, Ghosj, & Wells, 1994]
- printed wire assembly [Feo, Bard, & Holland, 1995; Bard, Feo, & Holland, 1996]

Scheduling (continued)

- single machine with sequence dependent setup costs & delay penalties [Feo, Sarathy, & McGahan, 1996]
- field technician scheduling [Xu & Chiu, 1996, 1997]
- flow shop with setup costs [Ríos-Mercado & Bard, 1997, 1998]
- school timetabling [Drexl & Salewski, 1997; Rivera, 1998]

Scheduling (continued)

- bus-driver scheduling [Ramalhinho, Paixão, & Portugal, 1998]
- vehicle scheduling [Atkinson, 1998]
- job shop scheduling [R., Binato, Hery, & Loewenstern, 2000]

Routing

- vehicle routing with time windows [Kontoravdis & Bard, 1995]
- vehicle routing [Hjorring, 1995]
- aircraft routing [Argüello, Bard, & Yu, 1997]
- inventory routing problem with satellite facilities [Bard et al., 1998]
- Vehicle routing with backhauls [Carreto & Baker, 2000]

Logic

- SAT [R. & Feo, 1996]
- MAX-SAT [Pardalos, Pitsoulis, & R., 1996, 1997, 1998]
- inferring logical clauses from examples [Deshpande & Triantaphyllou, 1998]

Partitioning

- graph two-partition [Laguna, Feo, & Elrod, 1994]
- number partitioning [Argüello, Feo, & Goldschmidt, 1996]
- circuit partitioning [Areibi & Vannelli, 1997; Areibi, 1999]

Location and Layout

- *p* - hub location [Klincewicz, 1992]
- pure integer capacitated plant location [Delmaire et al., 1997]
- location with economies of scale [Holmqvist, Migdalas, & Pardalos, 1997]
- traffic concentrator [R. & Ulular, 1997]
- single source capacitated plant location [Díaz & Fernández, 1998]
- maximum covering [R., 1998]
- dynamic facility layout [Urban, 1998]
- uncapacitated location problem [Gomes & Silva, 1999]

Graph theoretic

- max independent set [Feo, R., & Smith, 1994; R., Feo, & Smith, 1998]
- max clique with weighted edges [Macambira & Souza, 1997]
- graph planarization [R. & Ribeiro, 1997; Ribeiro & R., 1997]
- 2-layer straight line crossing minimization [Laguna & Martí, 1999]
- sparse graph coloring [Laguna & Martí, 1998]

Graph theoretic (continued)

- maximum weighted edge subgraph [Macambira & Meneses, 1998]
- Steiner problem [Martins, Pardalos, R., & Ribeiro, 1998; Martins & Ribeiro, 1998; Martins, R., & Ribeiro, 1999; Martins, R., Ribeiro, & Pardalos, 2000]
- feedback vertex set [Qian, Pardalos, & R., 1998; Festa, Pardalos, & R., 1999]
- maximum clique [Abello, Pardalos, & R., 1998; Pardalos, R., & Rappe, 1998]
- capacitated minimum spanning tree [Ahuja, Orlin, & Sharma, 1998]

Graph theoretic (continued)

- traveling salesman [Silveira, 1999]
- maximum cut [Festa, Pardalos, & R., 2000]

QAP & other assignment problems

- QAP [Li, Pardalos, & R., 1994]
- parallel GRASP for QAP [Pardalos, Pitsoulis, & R., 1995]
- Fortran subroutines for dense QAPs [R., Pardalos, & Li, 1996]
- initial population for GA for QAP [Ahuja, Orlin, & Tiwari, 2000]
- long term memory GRASP for QAP [Fleurent & Glover, 1999]
- biquadratic assignment problem [Mavridou, Pardalos, Pitsoulis, & R., 1997]

QAP & other assignment problems (continued)

- Fortran subroutines for sparse QAPs [Pardalos, Pitsoulis, & R., 1997]
- multidimensional knapsack [Labat & Mynard, 1997]
- data association multidimensional assignment problem [Murphey, Pardalos, & Pitsoulis, 1998]
- multidimensional assignment problem [Robertson, 1998]
- modified local search in GRASP for QAP [Rangel, Abreu, Boaventura-Netto, & Boeres, 1998]

QAP & other assignment problems (continued)

- 3-index assignment problem [Aiex, Pardalos, R., & Toraldo, 2000]

Miscellaneous problems

- set covering [Feo & R., 1989]
- concave-cost network flow problem [Holmqvist, Migdalas, & Pardalos, 1998]
- maximum diversity [Ghosj, 1996]
- protein folding [Krasnogor et al., 1998]
- clustering [Areibi & Vannelli, 1997; Areibi, 1999]
- consumer choice in competitive location models [Colomé & Serra, 1998]
- time series analysis [Medeiros, R., & Veiga, 1999; Medeiros, Veiga, & R., 1999]

Survey of industrial applications in literature

- manufacturing
- transportation
- telecommunications
- automatic drawing
- electrical power systems
- VLSI design
- military

Manufacturing

- discrete parts [Bard & Feo, 1989]
- cutting path & tool selection [Feo & Bard, 1989]
- equipment selection [Bard & Feo, 1991]
- component grouping [Klincewicz & Rajan, 1994]
- printed wiring board assembly [Feo, Bard, & Holland, 1995; Bard, Feo, & Holland, 1996]

Transportation

- flight scheduling & maintenance base planning [Feo & Bard, 1989]
- intermodal trailer assignment [Feo & González-Velarde, 1995]
- aircraft routing in response to groundings & delays [Argüello, Bard, & Yu, 1997]
- rail car unloading [Bard, 1997]
- airline fleet assignment [Sosnowska, 1999]

Telecommunications

- design of SDH mesh-restorable networks [Poppe, Pickavet, Arijs, & Demeester, 1997]
- Steiner tree in graphs [Martins, Pardalos, R., & Ribeiro, 1998; Martins & Ribeiro, 1998; Martins, R., & Ribeiro, 1999]
- permanent virtual circuit (PVC) routing [Resende & R., 1997; Resende & R., 1999; Festa, Resende, & R., 2000]
- traffic scheduling in satellite switched time division multi-access (SS/TDMA) systems [Prais & Ribeiro, 1998]

Telecommunications
(continued)

- point of presence (PoP) location [R., 1998]
- frequency assignment [Pasiliao, 1998; Liu, Pardalos, Rajasekaran, & R., 1999; Oliveira, Gomes, & R., 2000]

Automatic drawing

- seam drawing in mosaicking of aerial photographic maps [Fernández & Martí, 1997]
- graph planarization [R. & Ribeiro, 1997; Ribeiro & R., 1997]
- 2-layer straight line crossing minimization [Laguna & Martí, 1999]

Electrical power systems

- transmission expansion planning
  [Binato, Oliveira, & Araújo, 1998; Binato & Oliveira, 1999]

VLSI design

- circuit partitioning [Areibi & Vannelli, 1997; Areibi, 1999]

Military

- multitarget multisensor tracking
  [Murphey, Pardalos, & Pitsoulis, 1998]

Conclusion

- Online at my web site:
  - Up-to-date survey of GRASP [R., 1998]:

    http://www.research.att.com/~mgcr/doc/sgrasp.ps
  - Up-to-date bibliography:

    http://www.research.att.com/~mgcr/doc/graspbib.ps.Z

    http://www.research.att.com/~mgcr/doc/graspbib.bib
  - Up-to-date annotated bibliography [Festa & R., 2000]:

    http://www.research.att.com/~mgcr/doc/gabib.pdf